3-0155-SM

IMMEDIATE GENERATING METHOD

Patent Number: JP8194626
Publication date: 1996-07-30
Inventor(s): ISOBE TATSUO; KASHIWAGI YUUGO
Applicant(s):: HITACHI LTD; HITACHI SOFTWARE ENG CO LTD
Requested Patent: JP8194626
Application Number: JP19950022311 19950117
Priority Number(s):
IPC Classification: G06F9/45
EC Classification:
Equivalents:

Abstract

PURPOSE: To enable the designation of an immediate exceeding a limit without adding any instruction for immediate generation.
CONSTITUTION: An immediate #Z not exceeding a first instruction with an immediate #X exceeding a designated frame as an operand is defined as a first operand and a register held value Y remaining in any register is defined as a second operand for a four-rules arithmetic instruction and the immediate is generated by being replaced with this rule. Inside a storage device for dividing series of instructions into plural basic blocks so as to be the operating environment of a compiler, an immediate generating memory is provided with an operand information table on the operands of instructions consisting of the respective basic blocks, flow information table on the set and reference of the register and its order and register hold value table on the hold value of the register and when registers RA and RB for storing the result of the first instruction and the result of the arithmetic operations instruction are not equal, the registers RA and RB are replaced as operands after flow analysis of these registers is performed.

(19)日本国特許庁（ＪＰ）　(12) 公開特許公報（Ａ）　(11)特許出願公開番号

特開平8－194626

(43)公開日　平成８年(1996)７月30日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　9/45					
		7737－5Ｂ	Ｇ０６Ｆ　9/44	３２２　Ｅ	
		7737－5Ｂ		３２２　Ｈ	

審査請求　未請求　請求項の数４　ＦＤ　（全 13 頁）

(21)出願番号　特願平7－22311

(22)出願日　平成７年(1995)１月17日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地

(71)出願人　000233055
日立ソフトウエアエンジニアリング株式会社
神奈川県横浜市中区尾上町６丁目81番地

(72)発明者　磯部　竜雄
神奈川県横浜市中区尾上町６丁目81番地
日立ソフトウェアエンジニアリング株式会社内

(74)代理人　弁理士　徳若　光政

最終頁に続く

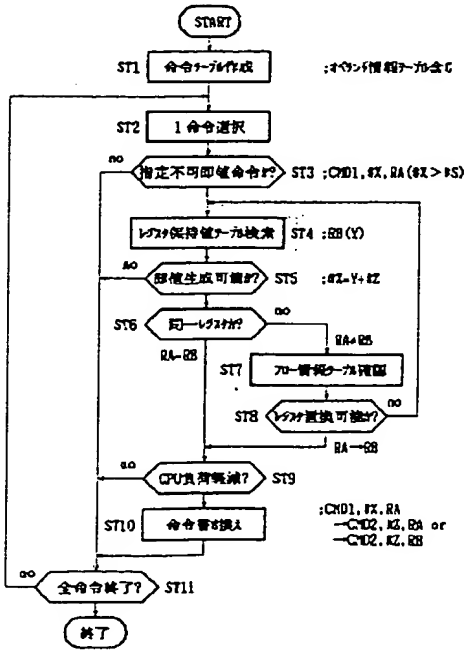
(54)【発明の名称】　即値生成方法

(57)【要約】　（修正有）

【目的】　即値生成のための命令追加をすることなく、制限を超えた即値の指定を可能にする。

【構成】　指定枠を超える即値＃Ｘをオペランドとする第１の命令を超えない即値＃Ｚを第１オペランドとしいずれかのレジスタに残存するレジスタ保持値Ｙを第２オペランドとする四則演算命令に置き換え生成する。一連の命令を、複数の基本ブロックに分割し、コンパイラの動作環境となる記憶装置内に、各基本ブロックを構成する命令のオペランドに関するオペランド情報テーブルと、レジスタの設定及び参照ならびにその順序に関するフロー情報テーブルと、レジスタの保持値に関するレジスタ保持値テーブルとを含む即値生成メモリを設け、第１の命令の結果と四則演算命令の結果が格納されるレジスタＲＡ及びＲＢとが同一のものでないときには、これらをフロー解析した上で、オペランドとしてのレジスタＲＡ及びＲＢを置き換える。

図４　コンパイラの即値生成処理フロー



【特許請求の範囲】

【請求項１】　原始プログラムを、所定数のレジスタを具備しかつその命令の少なくともオペランド部が固定長とされるプロセッサで実行可能なオブジェクトプログラムに変換する翻訳システムにおいて、上記オペランド部の指定枠を超える第１の即値として与えられたオペランドを、上記オペランド部の指定枠を超えない第２の即値と上記レジスタのいずれかに残存するレジスタ保持値とをもとに等価的に生成することを特徴とする即値生成方法。

【請求項２】　上記第１の即値を含む第１の命令は、第１の即値を第１オペランドとし第１のレジスタを第２オペランドとするものであり、上記レジスタ保持値は、第２のレジスタに残存するものであって、上記第１の即値として与えられたオペランドの生成は、上記第１の命令を、上記第２の即値を第１オペランドとし第２のレジスタを第２オペランドとする四則演算命令に置き換えることにより実現されるものであることを特徴とする請求項１の即値生成方法。

【請求項３】　上記第１の命令の結果は、上記第１のレジスタに格納され、上記四則演算命令の結果は、上記第２のレジスタに格納されるものであって、上記翻訳システムは、第１及び第２のレジスタが同一のものでないとき、これらのレジスタの以後の振る舞いをフロー解析し、必要に応じて以後の命令の第１又は第２のオペランドとして指定されている第１のレジスタを第２のレジスタに置き換えるものであることを特徴とする請求項２の即値生成方法。

【請求項４】　上記翻訳システムにおいて、一連の命令は、その中間に分岐の始点又は終点を含まない範囲を一つの単位として複数の基本ブロックに分割されるものであって、上記翻訳システムは、その動作環境の一部となる記憶装置内に、上記基本ブロックのそれぞれを構成する一つ又は複数の命令のオペランドに関するオペランド情報テーブルと、上記所定数のレジスタの設定及び参照ならびにその順序に関するフロー情報テーブルと、上記所定数のレジスタの保持値に関するレジスタ保持値テーブルとを含む即値生成メモリを備えるものであることを特徴とする請求項１，請求項２又は請求項３の即値生成方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】この発明は即値生成方法に関し、例えば、原始プログラムをオペランドとして与えうる即値の大きさに制限のあるプロセッサで実行可能なオブジェクトプログラムに変換するためのコンパイラ等の翻訳システムならびにその実行効率の向上に利用して特に有効な技術に関するものである。

【０００２】

【従来の技術】高級言語で書かれた原始プログラムを、マイクロコンピュータ等のプロセッサで実行可能な機械語のオブジェクトプログラムに変換するコンパイラ等の翻訳システムがある。また、命令回数や命令の種類を削減することにより、高頻度の処理を集中的に効率良く実行しうるＲＩＳＣ（Ｒｅｄｕｃｅｄ　Ｉｎｓｔｒｕｃｔｉｏｎ　Ｓｅｔ　Ｃｏｍｐｕｔｅｒ）等のコンピュータシステムがある。これらのコンピュータシステムでは、その命令の特にオペランド部が固定長とされ、オペランドとして与えうる即値の大きさに制限が設けられる場合がある。

【０００３】

【発明が解決しようとする課題】原始プログラムをオペランドとして与えうる即値の大きさに制限のあるプロセッサで実行可能なオブジェクトプログラムに変換する従来の翻訳システムでは、指定枠を超えた即値をオペランドとして指定したい場合、この即値を定数定義してラベルを付与した後、所望のレジスタにロードするか、指定枠内にある複数の即値を四則演算し又は連結することで指定枠外の即値を算出した後、所望のレジスタに格納するいわゆる命令生成法によらざるを得ない。しかし、前者の方法では、定数定義やラベル付与のためにメモリの有効領域が余分に費やされるとともに、ラベル参照に際して実質的なサイクルタイムの長いメモリアクセスが必要となる。また、後者の方法では、追加された即値演算命令の格納にメモリの有効領域が余分に費やされるとともに、オブジェクトプログラムに従って動作する実行用プロセッサのステップ数が増大する。この結果、いずれにおいても実行用プロセッサの処理負担が増加し、その実行効率が低下するものである。

【０００４】この発明の目的は、定数定義によるラベル付与又はラベル参照のためのメモリアクセスあるいは即値生成命令の追加を必要とすることなく、指定枠を超える大きさの即値の指定を可能にしうる即値生成方法を提供することにある。この発明の他の目的は、その命令が固定長とされる実行プロセッサの処理負担を軽減してその実行効率を高め、メモリの使用効率を高めることにある。

【０００５】この発明の前記ならびにその他の目的と新規な特徴は、この明細書の記述及び添付図面から明らかになるであろう。

【０００６】

【課題を解決するための手段】本願において開示される発明のうち代表的なものの概要を簡単に説明すれば、次の通りである。すなわち、原始プログラムを、所定数のレジスタを具備しかつその命令の少なくともオペランド部が固定長とされるプロセッサで実行可能なオブジェクトプログラムに変換する翻訳システムにおいて、指定枠を超える第１の即値として与えられたオペランドを、指定枠を超えない第２の即値といずれかのレジスタに残存するレジスタ保持値とをもとに、しかも第１の即値をオ

ペランドとする第１の命令を、第２の即値を第１のオペランドとしレジスタ保持値を第２のオペランドとする四則演算命令に置き換えることにより生成する。このため、一連の命令を、その中間に分岐の始点又は終点を含まない範囲を一つの単位として複数の基本ブロックに分割し、翻訳システムの動作環境となる記憶装置内に、各基本ブロックを構成する一つ又は複数の命令のオペランドに関するオペランド情報テーブルと、レジスタの設定及び参照ならびにその順序に関するフロー情報テーブルと、レジスタの保持値に関するレジスタ保持値テーブルとを含む即値生成メモリを設けるとともに、第１の命令の結果が格納される第１のレジスタと四則演算命令の結果が格納される第２のレジスタとが同一のものでないときは、これらのレジスタの以後の振る舞いを即値生成メモリによりフロー解析し問題がないことを確認した上で、第１及び第２のレジスタを置き換える。

【０００７】

【作用】上記した手段によれば、プログラムの実質的な機能を損いまた実行プロセッサの不正実行を招くことなく、しかも定数定義によるラベル付与又はラベル参照のためのメモリアクセスあるいは即値演算命令の追加を必要とすることなく、指定枠を超える大きさの即値の指定を可能としうる即値生成方法を実現することができる。この結果、その命令が固定長とされる実行プロセッサの処理負担を軽減してその実行効率を高め、メモリの使用効率を高めることができる。

【０００８】

【実施例】図１には、この発明が適用されたコンパイラ（翻訳システム）の動作環境を説明するための一実施例のシステム構成図が示されている。また、図２及び図３には、図１のコンパイラで用いられる即値生成メモリの一実施例の記憶領域構成図が示されている。これらの図をもとに、まずこの実施例のコンパイラの動作環境と即値生成メモリの記憶領域構成ならびにその特徴について説明する。

【０００９】図１において、この実施例のコンパイラは、中央処理装置（ＣＰＵ）１１を基本構成要素とする翻訳用コンピュータ１をその稼動環境とする。中央処理装置１１には、所定の記憶容量を有する記憶装置１２と、オペレータ及びコンピュータ間のマンマシンインタフェースとなるキーボード１３とが結合される。また、記憶装置１２には、コンパイラつまり翻訳プログラム４を格納するための領域と、この翻訳プログラム４による翻訳処理の過程で用いられる即値生成メモリ５の領域とが設けられる。翻訳用コンピュータ１は、所定の高級言語で書かれた原始プログラム３を、翻訳プログラム４に従って、実行用マイコン（マイクロコンピュータ）２で実行可能な機械語のオブジェクトプログラム６に変換する。

【００１０】一方、オブジェクトプログラム６の実行環境となる実行用マイコン２は、中央処理装置２１を基本構成要素とし、この中央処理装置２１には、所定の記憶容量を有するＲＯＭ（リードオンリーメモリ）２２が結合される。ＲＯＭ２２には、コンパイラにより作成されたオブジェクトプログラム６が実行プログラム７として格納される。実行用マイコン２は、ＲＯＭ２２内の実行プログラム７により制御され、図示されない入出力装置の制御や所定の演算処理を行う。

【００１１】この実施例において、実行用マイコン２を構成するプロセッサつまり中央処理装置２１は、ｐ＋１個の汎用のレジスタＲ０～Ｒｐとユーザによって定義可能なｑ＋１個のスペシャルレジスタＳＰ０～ＳＰｑとを備える。また、その命令は固定長とされ、特にその第１及び第２オペランド部はそれぞれ例えば８ビットとされる。したがって、命令の第１及び第２オペランドとして指定しうる即値の大きさは、－１２８から＋１２７までの範囲に制限される。これに対処するため、この実施例のコンパイラは、指定枠を超えた即値として指定された第１オペランドを、指定枠を超えない他の即値といずれかのレジスタに残存するレジスタ保持値とをもとに等価的に生成する即値生成機能を有し、その動作環境となる記憶装置１２には、即値生成処理に供される即値生成メモリ５が設けられる。なお、コンパイラの具体的な即値生成方法については、後で詳細に説明する。

【００１２】記憶装置１２内に設けられた即値生成メモリ５は、図２及び図３に示されるように、命令ポインタ５１，レジスタ探索ポインタ５２，命令テーブル５３，オペランド情報テーブル５４，フロー情報テーブル５５，レジスタ保持値テーブル５６，レジスタフラグテーブル５７，スペシャルレジスタ保持値テーブル５８及びスペシャルレジスタフラグテーブル５９を含む。この実施例において、プログラムを構成する一連の命令は、その中間に分岐の始点又は終点を含まない範囲を一つの単位として所定数の基本ブロック５０つまり５００及び５０１等に分割される。命令ポインタ５１，レジスタ探索ポインタ５２，レジスタ保持値テーブル５６，レジスタフラグテーブル５７，スペシャルレジスタ保持値テーブル５８及びスペシャルレジスタフラグテーブル５９は、すべての基本ブロック５０に共通に設けられ、命令テーブル５３，オペランド情報テーブル５４及びフロー情報テーブル５５は、各基本ブロック５０に対応して個別に設けられる。

【００１３】ここで、即値生成メモリ５の命令テーブル５３は、各基本ブロックの命令テーブルを順次リンクさせるためのリンク情報ＬＩＮＫ０及びＬＩＮＫ１等と、その中間に分岐の始点又は終点を含まない任意数の命令ＩＮＳＴ００～ＩＮＳＴ０３等ならびにＩＮＳＴ１０～ＩＮＳＴ１３等とを保持する。また、オペランド情報テーブル５４は、命令テーブル５３の対応する命令ＩＮＳＴ００～ＩＮＳＴ０３等ならびにＩＮＳＴ１０～ＩＮＳ

Ｔ１３等の第１オペランド５４１及び第２オペランド５４２を保持し、その内容は、レジスタＲ０～ＲｐあるいはスペシャルレジスタＳＲ０～ＳＲｑであったり、即値ＩＭＭＤであったりする。

【００１４】一方、フロー情報テーブル５５は、設定レジスタ情報ＲＥＧＳ　ＬＩＳＴ０及びＲＥＧＳ　ＬＩＳＴ１等，参照レジスタ情報ＲＥＧＲ　ＬＩＳＴ０及びＲＥＧＲ　ＬＩＳＴ１等，設定参照順序情報Ｓ／Ｒ　ＯＲＤ０及びＳ／Ｒ　ＯＲＤ１等ならびに分岐先情報ＢＲＡＮＣＨ０及びＢＲＡＮＣＨ１等を含む。このうち、設定レジスタ情報ＲＥＧＳ　ＬＩＳＴ０及びＲＥＧＳ　ＬＩＳＴ１等は、その各ビットがレジスタＲ０～ＲｐあるいはスペシャルレジスタＳＲ０～ＳＲｑに対応され、対応する基本ブロックにおいて対応するレジスタ又はスペシャルレジスタに何らかの値が設定されているとき選択的にセットされる。同様に、参照レジスタ情報ＲＥＧＲ　ＬＩＳＴ０及びＬＩＳＴＲ　ＬＩＳＴ１等は、その各ビットがレジスタＲ０～ＲｐあるいはスペシャルレジスタＳＲ０～ＳＲｑに対応され、対応する基本ブロックにおいて対応するレジスタ又はスペシャルレジスタがいずれかの命令によって参照されるとき選択的にセットされる。

【００１５】さらに、レジスタ保持値テーブル５６は、対応するレジスタＲ０～Ｒｐに設定され残存する値を保持する。また、レジスタフラグテーブル５７は、その各ビットがレジスタＲ０～Ｒｐに対応され、メモリ内容が転送又は演算の対象となりあるいはアドレスが設定されたために対応するレジスタＲ０～Ｒｐの保持値が翻訳時に未知の値となったとき、選択的にセットされる。同様に、スペシャルレジスタ保持値テーブル５８は、対応するスペシャルレジスタＳＲ０～ＳＲｑに設定され残存する値を保持する。また、スペシャルレジスタフラグテーブル５８は、その各ビットがスペシャルレジスタＳＲ０～ＳＲｑに対応され、メモリ内容が転送又は演算の対象となりあるいはアドレスが設定されたために対応するスペシャルレジスタＳＲ０～ＳＲｑの保持値が翻訳時に未知の値となったとき、選択的にセットされる。

【００１６】命令ポインタ５１は、後述するように、コンパイラによる翻訳処理の過程で、命令テーブル５３に格納された命令ＩＮＳＴ００～ＩＮＳＴ０３等ならびにＩＮＳＴ１０～ＩＮＳＴ１３等を順次指定し、指定枠を超えた即値を含む命令に対して即値生成処理を選択的に施すためのポインタとなる。また、レジスタ探索ポインタ５２は、この即値生成処理の過程で、レジスタ保持値テーブル５６ならびにスペシャルレジスタ保持値テーブル５８を順次指定し、指定枠を超えた即値を等価的に生成すべくレジスタ保持値を探索するためのポインタとなる。

【００１７】図４には、図１のコンパイラによる即値生成処理の一実施例のフロー図が示されている。また、図５には、図１のコンパイラの処理対象となるプログラムのうち即値生成が必要な部分の一実施例の部分的な命令リストが示され、図６には、そのもう一つの実施例の部分的な命令リストが示されている。これらの図をもとに、この実施例のコンパイラによる即値生成処理の具体的方法ならびにその特徴について説明する。なお、図５には、指定枠を超えた即値を含む命令の第２オペランドとなるレジスタ（第１のレジスタ）と即値生成に供されるレジスタ保持値を保持するレジスタ（第２のレジスタ）とが同一レジスタＲ１である場合が例示され、図６には、これらのレジスタが同一レジスタではなく異なるレジスタＲ１及びＲ２である場合が例示される。また、図５及び図６には、（ａ）として原始プログラムに近い翻訳処理の中間段階つまりは即値生成処理が施されない状態におけるオブジェクトプログラムが、（ｂ）として（ａ）のプログラムを従来のコンパイラで処理した後のオブジェクトプログラムが、また（ｃ）として（ａ）のプログラムをこの発明が適用されたコンパイラで即値生成処理した後のオブジェクトプログラムがそれぞれ示されている。さらに、各命令リストの左外側には、各命令のプログラム行番号が付記され、その右外側には、各命令に関するコメントが付記されているので、説明にあわせて参照されたい。

【００１８】図４において、この実施例のコンパイラによる即値生成処理は、ステップＳＴ１つまりオペランド情報テーブル５４を含む命令テーブル５３の作成によって開始される。このとき、翻訳過程にあるプログラムを構成する一連の命令は、前述のように、その中間に分岐の始点又は終点を含まない範囲を一つの単位として複数の基本ブロック５０に分割され、翻訳用コンピュータ１の記憶装置１２の即値生成メモリ５には、これらの基本ブロック５０に対応して作成された命令テーブル５３及びオペランド情報テーブル５４が用意される。

【００１９】命令テーブル５３及びオペランド情報テーブル５４の作成が終了すると、コンパイラは、ステップＳＴ２により、最初の基本ブロック５００から命令ポインタ５１によって指定される一つの命令を選択し、ステップＳＴ３により、選択された命令が指定枠を超えた即値を第１オペランドとして含むいわゆる指定不可即値命令であるかどうかを判定する。このとき、命令は、コメントとして付記されるように、コマンドＣＭＤ１と、第１オペランドとなる即値＃Ｘ（ここで、＃はオペランドがＸなる十進数の即値からなることを表す。以下同様）と、第２オペランドとなるレジスタＲＡとを含み、この命令が指定不可即値命令である場合、第１オペランドである即値＃Ｘはその指定枠＃Ｓより大きな値となる。この実施例において、実行用マイコン２が実行しうる命令の第１オペランドは、前述のように、８ビットとされ、その最大値＃Ｓは＋１２７とされる。なお、各命令で第２オペランドとして指定されるレジスタＲＡ等は、当該

命令の実行結果を格納するためのいわゆるデスティネーションオペランドともなる。

【００２０】ステップＳＴ３において、選択された命令が指定枠を超える即値を含まず指定不可即値命令ではない場合、コンパイラは、ステップＳＴ１１により、すべての命令に関する選択処理が終了したかを判定し、まだ未選択の命令が存在する場合には、ステップＳＴ２に戻り、未選択の命令が存在しない場合には、一連の即値生成処理を終結する。なお、コンパイラは、ステップＳＴ１１の判定処理に移行する直前で、図示されないフロー情報テーブル５５，レジスタ保持値テーブル５６，レジスタフラグテーブル５７，スペシャルレジスタ保持値テーブル５８及びスペシャルレジスタフラグテーブル５９の更新処理を行う。すなわち、コンパイラは、選択された命令によりレジスタＲ０～ＲｐあるいはスペシャルレジスタＳＲ０～ＳＲｑに何らかの値が設定され又は参照されるのをモニタし、フロー情報テーブル５５の設定レジスタ情報及び参照レジスタの対応するビットをセットするとともに、各レジスタに設定された値を可能な限りレジスタ保持値テーブル５６又はスペシャルレジスタ保持値テーブル５８に登録する。また、選択された命令によりメモリ内容が転送又は演算の対象となりあるいはアドレスが設定されたために対応するレジスタＲ０～Ｒｐあるいはスペシャルレジスタ ＳＲ０～ＳＲｑの保持値が不定となったときには、レジスタフラグテーブル５７又はスペシャルレジスタフラグテーブル５９の対応するビットをセットする。

【００２１】一方、ステップＳＴ３において、選択された命令が指定枠を超える即値＃Ｘ（第１の即値）を含む指定不可即値命令であった場合、コンパイラは、ステップＳＴ４により、レジスタ保持値テーブル５６を参照し、その保持値Ｙと指定枠を超えない即値＃Ｚ（第２の即値）とにより指定枠を超えた即値＃Ｘを等価的に生成できそうなレジスタを探索する。すなわち、例えば、図５（ａ）のプログラム行Ｌ２３の移動命令『ＭＯＶ　＃１２８，Ｒ１』のように、選択された命令の第１オペランドが指定枠を超える即値＃Ｘつまり“１２８”であった場合、コンパイラは、その前に実行された移動命令『ＭＯＶ　＃１２７，Ｒ１』によりレジスタＲ１に設定された指定枠内の値Ｙつまり“１２７”に着目する。そして、

１２８－１２７＝１

であることから、指定枠内の即値＃Ｚとして、

＃Ｚ＝１

を算出する。この結果、ステップＳＴ５において、

＃Ｘ＝Ｙ＋＃Ｚ

つまり、例えば『ＡＤＤ　＃Ｚ，Ｒ１』なる加算命令により“１２８”なる指定枠外の即値＃Ｘが等価的に生成可能であることを判定する。

【００２２】即値生成可能であることを判定したコンパイラは、ステップＳＴ６により、もとの命令『ＣＭＤ１　＃Ｘ，ＲＡ』の第２オペランドつまりデスティネーションオペランドとなるレジスタＲＡ（第１のレジスタ）と、レジスタ保持値Ｙを保持するレジスタＲＢ（第２のレジスタ）とが同一レジスタであるかどうかを判定する。その結果、レジスタＲＡ及びＲＢが同一レジスタである場合には、ステップＳＴ９において、命令置き換えにより実行用マイコン２の中央処理装置２１の処理負荷が軽減されるかどうかを判定し、負荷軽減の見込みがある場合には、ステップＳＴ１０により、もとの命令『ＣＭＤ１　＃Ｘ，ＲＡ』（第１の命令）を新しい命令『ＣＭＤ２　＃Ｚ，ＲＡ』（第２の命令）に置き換える。これにより、例えば図５（ａ）のプログラム行Ｌ２３の移動命令『ＭＯＶ　＃１２８，Ｒ１』は、図５（ｃ）のプログラム行Ｌ４３の加算命令『ＡＤＤ　＃１，Ｒ１』にそのまま１対１で置き換えられ、同一の実行結果が得られるものとなる。

【００２３】なお、ステップＳＴ６において、レジスタＲＡ及びＲＢが同一レジスタでないと判定された場合、コンパイラは、ステップＳＴ７により、フロー情報テーブル５５の設定レジスタ情報及び参照レジスタ情報ならびに設定参照順序情報を参照し、ステップＳＴ８により、レジスタＲＡ及びＲＢの置き換えが可能であるかどうかを判定する。すなわち、コンパイラは、フロー情報テーブル５５の設定レジスタ情報及び参照レジスタ情報ならびに設定参照順序情報をもとに、レジスタＲＡ及びＲＢの以後の振る舞いをフロー解析する。そして、レジスタＲＡが、

（１）対応する命令以後の同一基本ブロック内あるいはフロー情報テーブル５５の分岐先情報ＢＲＡＮＣＨ０及びＢＲＡＮＣＨ１等をもとにたぐりうるすべての基本ブロック内で設定されていること

（２）次に設定されるまでに参照がないこと

（３）参照があったとしてもその基本ブロックに分岐するのは当該基本ブロックのみであること

なる条件を満たし、しかもレジスタＢが、

（１）対応する命令以後の同一基本ブロック内あるいはフロー情報テーブル５５の分岐先情報ＢＲＡＮＣＨ０及びＢＲＡＮＣＨ１等をもとにたぐりうるすべて基本ブロック内で設定されていること

（２）次に設定されるまでに参照がないこと

なる条件を満たすことを確認した上で、ステップＳＴ１０による命令の書き換えを実行するとともに、新しい命令『ＣＭＤ２　＃Ｚ，ＲＡ』の第２オペランドをレジスタＲＢに置き換えて『ＣＭＤ２　＃Ｚ，ＲＢ』とし、以後の命令でオペランドとして指定されるレジスタＲＡもレジスタＲＢに置き換える。なお、このフローの中に図示してはいないが、命令１個１個処理してゆくと同時にレジスタ保持値テーブルを逐次更新してゆく処理が必要なことは言うまでもない。

【００２４】これらのことから、例えば、図６（ａ）の指定枠を超える即値＃２５０を含むプログラム行Ｌ５３の移動命令『ＭＯＶ　＃２５０，Ｒ２』に着目した場合、プログラム行Ｌ５１の移動命令『ＭＯＶ　＃１２７，Ｒ１』の実行結果としてレジスタＲ１に設定された値“１２７”を利用して即値の生成は可能であるが、この値“１２７”を保持するレジスタＲ１がもとの移動命令『ＭＯＶ　＃２５０，Ｒ２』の第２オペランドとなるレジスタＲ２とは異なるため、これらのレジスタの以後の振る舞いをフロー解析した上で、もとの移動命令は図６（ｃ）のプログラム行Ｌ７３の加算命令『ＡＤＤ　＃１２３，Ｒ１』に置き換えられる。言うまでもなく、レジスタＲ１の保持値は“１２７”であり、この加算命令の結果“２５０”は同じレジスタＲ１に格納される。また、図６（ａ）のプログラム行Ｌ５４の減算命令『ＳＵＢ　Ｒ２，Ｒ３』の第１オペランドとして指定されるレジスタＲ２は、上記レジスタの置き換えを受けてレジスタＲ１に置き換えられる。この結果、プログラムの本来の機能を維持できるとともに、レジスタ置き換えにともなう実行用マイコン２の不正実行を防止することができる。

【００２５】ところで、即値生成機能を持たない従来のコンパイラでは、図５（ａ）の移動命令『ＭＯＶ　＃１２８，Ｒ１』あるいは図６（ａ）の移動命令『ＭＯＶ　＃２５０，Ｒ２』のような指定枠を超える即値を直接指定できない。したがって、このような即値＃１２８又は＃２５０を指定したい場合、図５（ｂ）のプログラム行Ｌ３５の定義命令『ＤＣＬ　１２８』あるいは図６（ｂ）の定義命令『ＤＣＬ２５０』によってこれを定義し、『ＬＡＢＥＬ１』又は『ＬＡＢＥＬ２』なるラベルを付与した後、もとの移動命令を図５（ｂ）のプログラム行３３の移動命令『ＭＯＶ　ＬＡＢＥＬ１，Ｒ１』あるいは図６（ｂ）のプログラム行６３の移動命令『ＭＯＶ　ＬＡＢＥＬ１，Ｒ１』に書き換えるか、もとの移動命令の直前に１２８又は２５０を結果とする四則演算命令を追加し、もとの移動命令の第１オペランドを四則演算命令のデスティネーションオペランドとなるレジスタに書き換える必要がある。しかし、定数定義を行う図５（ｂ）及び図６（ｂ）の方法では、ラベル付与のためにメモリの有効領域が余分に費やされるとともに、プログラム行Ｌ３３又はＬ６３のラベル参照に際して実質的なサイクルタイムの長いメモリアクセスが必要となり、四則演算命令を追加する方法では、追加命令の格納にメモリの有効領域が余分に費やされるとともに、実行用マイコン２におけるオブジェクトプログラム６のステップ数が増大する。この結果、いずれの場合も実行用マイコン２の処理負担が増大し、その実行効率が低下する。

【００２６】ところが、この発明が適用されたコンパイラの場合、指定枠を超えた即値を含む移動命令『ＭＯＶ　＃１２８，Ｒ１』及び『ＭＯＶ　＃２５０，Ｒ２』は、前述のように、指定枠を超えない即値＃１又は＃１２３とレジスタＲ１の保持値１２７とによる加算命令『ＡＤＤ　＃１，Ｒ１』又は『ＡＤＤ　＃１２３，Ｒ２』にそのまま１対１で置き換えられるとともに、これらの命令に関与するレジスタの以後の振る舞いがフロー解析され、必要に応じてオペランドの置き換えが行われる。これらのことから、プログラムの実質的な機能を損いまた実行用マイコン２の不正実行を招くことなく、しかも定数定義によるラベル付与又はラベル参照のためのメモリアクセスあるいは即値生成のための命令追加を必要とすることなく、指定枠を超える大きさの即値の指定を可能としうる即値生成方法を実現できる。この結果、その命令が固定長とされる実行用マイコン２の処理負担を軽減してその実行効率を高め、メモリの使用効率を高めることができる。

【００２７】以上の実施例から得られる作用効果は、下記の通りである。すなわち、

（１）原始プログラムを、所定数のレジスタを具備しかつその命令の少なくともオペランド部が固定長とされるプロセッサで実行可能なオブジェクトプログラムに変換する翻訳システムにおいて、指定枠を超える第１の即値として与えられたオペランドを、指定枠を超えない第２の即値といずれかのレジスタに残存するレジスタ保持値とをもとに、しかも第１の即値をオペランドとする第１の命令を、第２の即値を第１のオペランドとしレジスタ保持値を第２のオペランドとする四則演算命令に置き換えることにより生成することで、定数定義によるラベル付与又はラベル参照のためのメモリアクセスあるいは即値生成のための命令追加を必要とすることなく、指定枠を超える大きさの即値の指定を可能としうる即値生成方法を実現することができるという効果が得られる。

【００２８】（２）上記（１）項において、一連の命令を、その中間に分岐の始点又は終点を含まない範囲を一つの単位として複数の基本ブロックに分割し、翻訳システムの動作環境となる記憶装置内に、各基本ブロックを構成する命令のオペランドに関するオペランド情報テーブルと、レジスタの設定及び参照ならびにその順序に関するフロー情報テーブルと、レジスタの保持値に関するレジスタ保持値テーブルとを含む即値生成メモリを設けるとともに、第１の命令の結果が格納される第１のレジスタと四則演算命令の結果が格納される第２のレジスタとが同一のものでないときには、これらのレジスタの以後の振る舞いを即値生成メモリによりフロー解析し問題がないことを確認した上で、選択的に第１及び第２のレジスタを置き換えることで、プログラムの本来の機能を保ち、プロセッサ等の不正実行を防止することができるという効果が得られる。

（３）上記（１）項及び（２）項により、その命令が固定長とされるプロセッサ等の処理負担を軽減してその実行効率を高めることができるとともに、そのメモリの使

用効率を高めることができるという効果が得られる。

【００２９】以上、本発明者によってなされた発明を実施例に基づき具体的に説明したが、この発明は、上記実施例に限定されるものではなく、その要旨を逸脱しない範囲で種々変更可能であることは言うまでもない。例えば、図１において、翻訳用コンピュータ１及び実行用マイコン２のブロック構成は、種々の実施形態を採りうる。図２及び図３において、即値生成メモリ５の構成は、この実施例による制約を受けないし、その利用方法も任意である。図４において、コンパイラによる即値生成処理は、基本的なアルゴリズムが変わらないことを条件に、種々の処理フローを採りうる。図５及び図６において、命令の構成，オペランド等の組み合わせ及び順序等は、種々の実施形態が考えられよう。即値生成のための演算命令には、加算命令以外の四則演算命令を利用してもよいし、四則演算命令以外の命令を利用することもできる。また、実行用マイコン２は、必ずしもスペシャルレジスタＳＲ０～ＳＲｑを備えることを必須条件とはしない。さらに、即値生成の対象となるオペランドは、第１オペランドに限定されないし、即値の具体的な指定枠や命令の種別及び名称等も、この実施例の限りではない。

【００３０】以上の説明では、主として本発明者によってなされた発明をその背景となった利用分野であるコンパイラに適用した場合について説明したが、それに限定されるものではなく、例えばアセンブラやオプティマイザ等の各種翻訳システムにも適用できる。この発明は、少なくとも原始プログラムをオペランドとして与えうる即値の大きさに制限のあるプロセッサで実行可能なオブジェクトプログラムに変換するための翻訳システムならびにその即値生成方法に広く適用できる。

【００３１】

【発明の効果】本願において開示される発明のうち代表的なものによって得られる効果を簡単に説明すれば、下記の通りである。すなわち、原始プログラムを、所定数のレジスタを具備しかつその命令の少なくともオペランド部が固定長とされるプロセッサで実行可能なオブジェクトプログラムに変換する翻訳システムにおいて、指定枠を超える第１の即値として与えられたオペランドを、指定枠を超えない第２の即値といずれかのレジスタに残存するレジスタ保持値とをもとに、しかも第１の即値をオペランドとする第１の命令を、第２の即値を第１のオペランドとしレジスタ保持値を第２のオペランドとする四則演算命令に置き換えることにより生成する。このため、一連の命令を、その中間に分岐の始点及び終点を含まない範囲を一つの単位として複数の基本ブロックに分割し、翻訳システムの動作環境となる記憶装置内に、各基本ブロックを構成する命令のオペランドに関するオペランド情報テーブルと、レジスタの設定及び参照ならびにその順序に関するフロー情報テーブルと、レジスタの保持値に関するレジスタ保持値テーブルとを含む即値生成メモリを設けるとともに、第１の命令の結果が格納される第１のレジスタと四則演算命令の結果が格納される第２のレジスタが同一のものでないときには、これらのレジスタの以後の振る舞いを即値生成メモリによりフロー解析し問題がないことを確認した上で、第１及び第２のレジスタを置き換える。これにより、プログラムの本来の機能を損いまたプロセッサの不正実行を招くことなく、しかも定数定義によるラベル付与又はラベル参照のためのメモリアクセスあるいは即値生成のための命令追加を必要とすることなく、指定枠を超える大きさの即値の指定を可能としうる即値生成方法を実現することができる。この結果、その命令が固定長とされるプロセッサ等の処理負担を軽減してその実行効率を高め、メモリの使用効率を高めることができる。

【図面の簡単な説明】

【図１】この発明が適用された即値生成方法を採るコンパイラの動作環境を説明するための一実施例を示すシステム構成図である。

【図２】図１のコンパイラで用いられる即値生成メモリの一実施例を示す部分的な記憶領域構成図である。

【図３】図１のコンパイラで用いられる即値生成メモリの一実施例を示す他の部分的な記憶領域構成図である。

【図４】図１のコンパイラによる即値生成処理の一実施例を示すフロー図である。

【図５】図１のコンパイラの処理対象となるプログラムのうち即値生成を必要とする部分の一実施例を示す命令リストである。

【図６】図１のコンパイラの処理対象となるプログラムのうち即値生成を必要とする部分の他の一実施例を示す命令リストである。

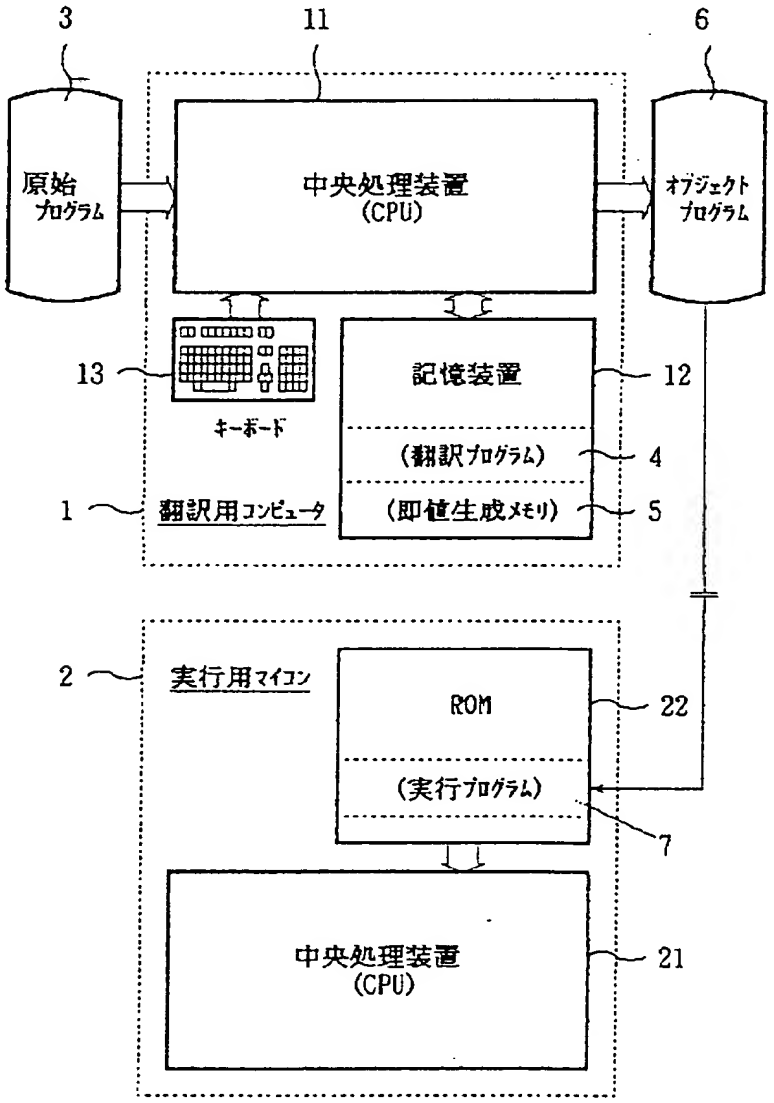
【符号の説明】

１・・・翻訳用コンピュータ、２・・・実行用マイコン（マイクロコンピュータ）、１１，２１・・・中央処理装置、１２・・・記憶装置、２２・・・ＲＯＭ（リードオンリーメモリ）、１３・・・キーボード、３・・・原始プログラム、４・・・翻訳プログラム（コンパイラ）、５・・・即値生成メモリ、６・・・オブジェクトプログラム、７・・・実行プログラム。５０，５００～５０１・・・基本ブロック、５１・・・命令ポインタ、５２・・・レジスタ探索ポインタ５３・・・命令テーブル、ＬＩＮＫ０～ＬＩＮＫ１・・・リンク情報、ＩＮＳＴ００～ＩＮＳＴ０３ないしＩＮＳＴ１０～ＩＮＳＴ１３・・・命令、５４・・・オペランド情報テーブル、５４１・・・第１オペランド、５４２・・・第２オペランド、ＩＭＭＤ・・・即値、Ｒ０～Ｒｐ・・・レジスタ、ＳＲ０～ＳＲｑ・・・スペシャルレジスタ、５５・・・フロー情報テーブル、ＲＥＧＳ　ＬＩＳＴ０，ＲＥＧＳ　ＬＩＳＴ１・・・設定レジスタ情報、ＲＥＧＲ　ＬＩＳＴ０，ＲＥＧＲ　ＬＩＳＴ１・・・参照レジスタ情

報、S／RORD0，S／R　ORD1・・・設定参照順序情報、BRANCH0，BRANCH1・・・分岐先情報。56・・・レジスタ保持値テーブル、REG0～REGp・・・レジスタ保持値、57・・・レジスタフラグテーブル、REGF　LIST・・・レジスタフラグリスト、58・・・スペシャルレジスタ保持値テーブル、SPE　REG0～SPE　REGq・・・スペシャルレジスタ保持値、59・・・スペシャルレジスタフラグテーブル、SPE　REGF　LIST・・・スペシャルレジスタフラグ情報。ST1～ST11・・・処理ステップ、CMD1～CMD2・・・コマンド、RA～RB・・・レジスタ、#S，#X，#Y，#Z・・・即値。L21～L24ないしL71～L74・・・プログラム行。

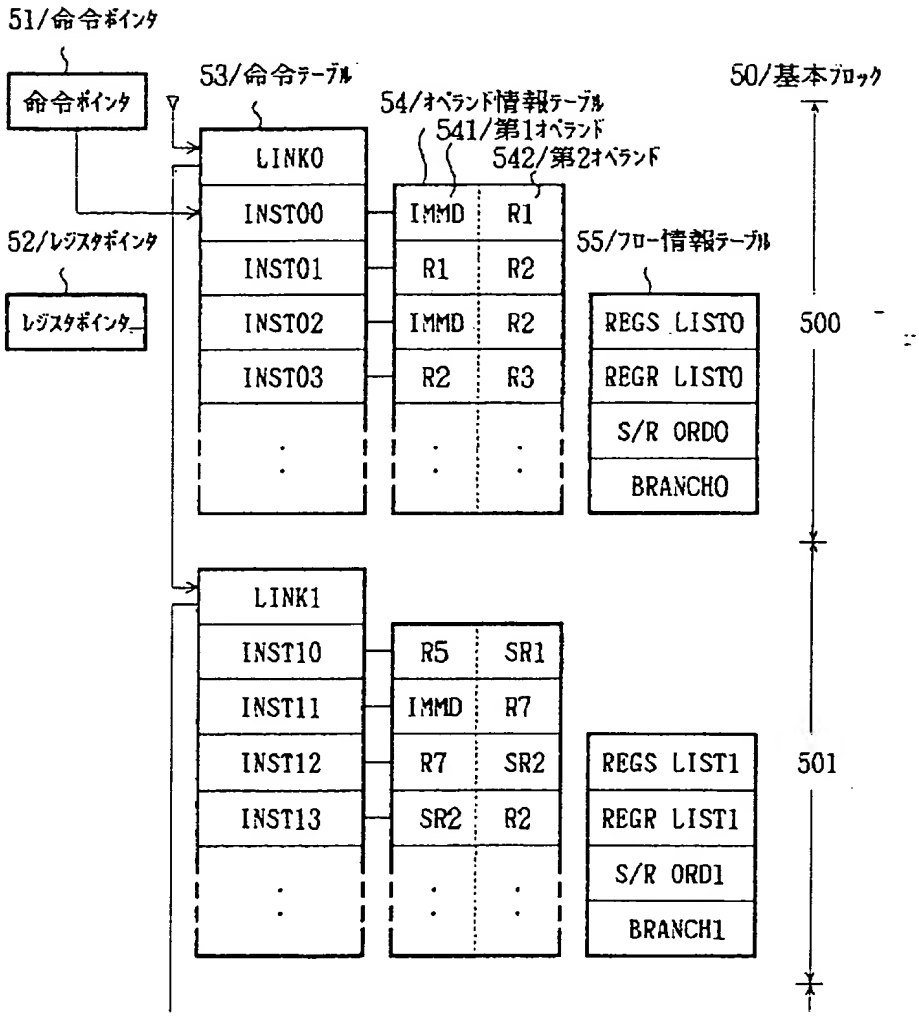
【図1】

図1　コンパイラの動作環境



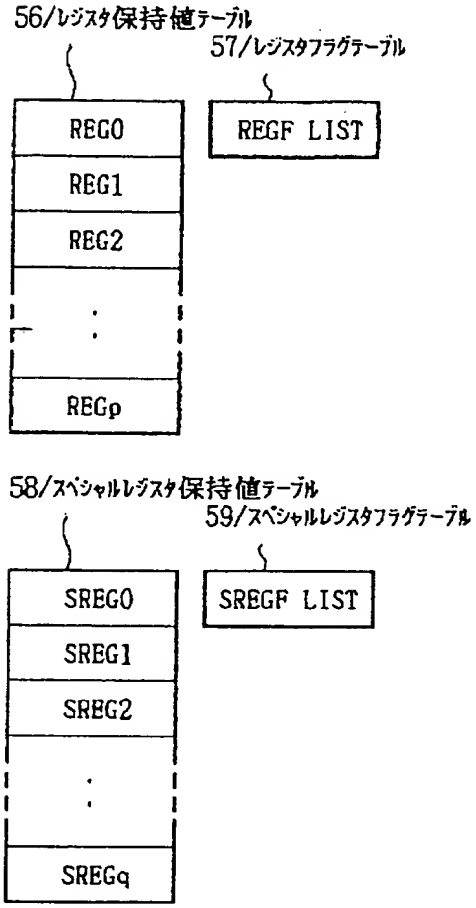
【図２】

図２　即値生成メモリの部分的な記憶領域構成（その１）



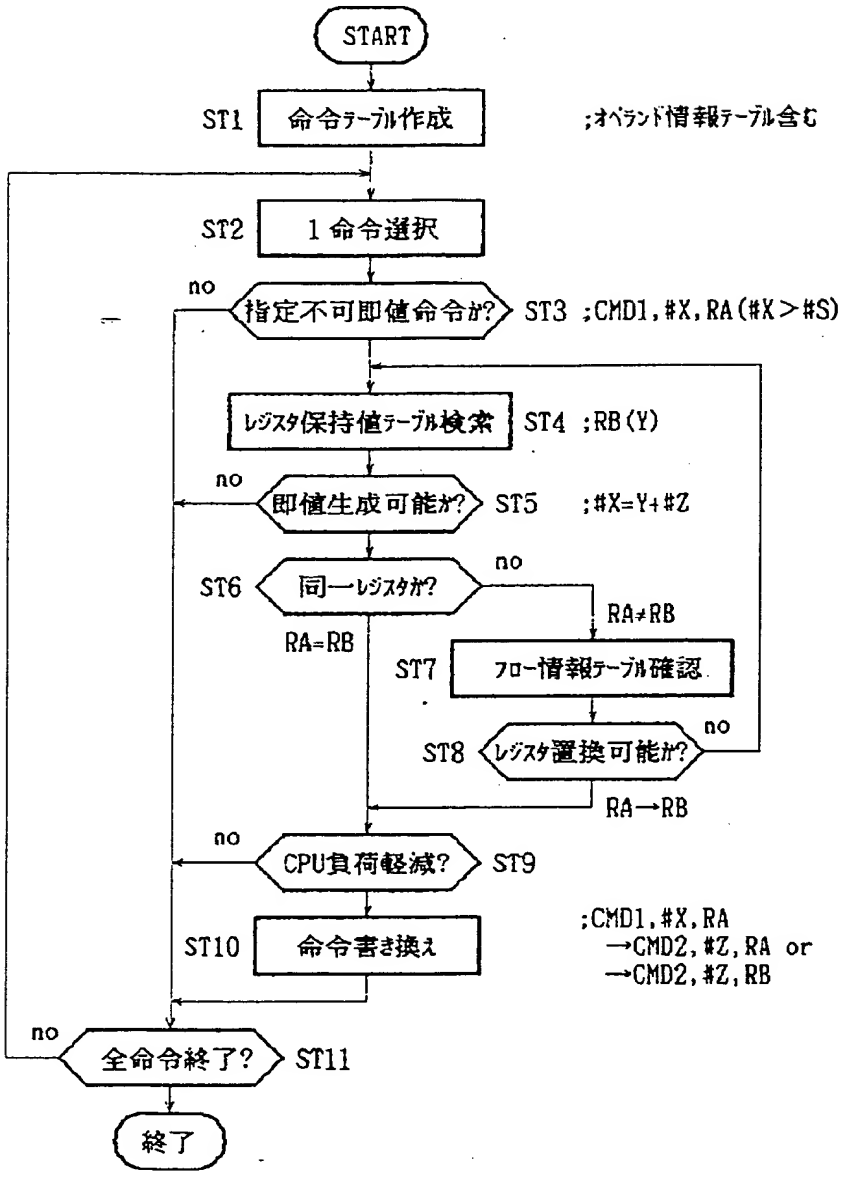
【図３】

図３　即値生成メモリの部分的な記憶領域構成（その２）



【図４】

図４ コンパイラの即値生成処理フロー



【図５】

図５　即値生成が必要なプログラムの一例（レジスター致）

（ａ）翻訳の中間段階におけるオブジェクトプログラム

```
:
L21     MOV   #127  ,R1    ;127→R1
L22     ADD   R1    ,R3    ;R1+R3→R3
:
L23     MOV   #128  ,R1    ;128→R1/指定枠ｵｰﾊﾞｰ
L24     SUB   R1    ,R3    ;R1-R3→R3
:
```

（ｂ）従来のコンパイラによるオブジェクトプログラム

```
:
L31     MOV   #127  ,R1    ;127→R1
L32     ADD   R1    ,R3    ;R1+R3→R3
:
L33     MOV   LABEL1,R1    ;LABEL1→R1/ﾒﾓﾘ参照発生
L34     SUB   R1    ,R3    ;R1-R3→R3
:
L35  LABEL1:DCL   128      ;LABEL1の定義
:
```

（ｃ）本発明のコンパイラによるオブジェクトプログラム

```
:
L41     MOV   #127  ,R1    ;127→R1
L42     ADD   R1    ,R3    ;R1+R3→R3
:
L43     ADD   #1    ,R1    ;1+R1→R1/命令置き換え
L44     SUB   R1    ,R3    ;R1-R3→R3
:
```

【図６】

図６　即値生成が必要なプログラムの一例（レジスタ不一致）

（ａ）翻訳の中間段階におけるオブジェクトプログラム

```
:
L51     MOV   #127  ,R1     ;127→R1
L52     ADD   R1    ,R3     ;R1+R3→R3
:
L53     MOV   #250  ,R2     ;250→R2/指定枠ｵｰﾊﾞｰ
L54     SUB   R2    ,R3     ;R2-R3→R3
:
```

（ｂ）従来のコンパイラによるオブジェクトプログラム

```
:
L61     MOV   #127  ,R1     ;127→R1
L62     ADD   R1    ,R3     ;R1+R3→R3
:
L63     MOV   LABEL2,R2     ;LABEL2→R2/ﾒﾓﾘ参照発生
L64     SUB   R2    ,R3     ;R2-R3→R3
:
L65  LABEL2:DCL   250       ;LABEL2の定義
:
```

（ｃ）本発明のコンパイラによるオブジェクトプログラム

```
:
L71     MOV   #127  ,R1     ;127→R1
L72     ADD   R1    ,R3     ;R1+R3→R3
:
L73     ADD   #123  ,R1     ;123+R1→R1/命令書き換え
L74     SUB   R1    ,R3     ;R1-R3→R3/ｵﾍﾟﾗﾝﾄﾞ書き換え
:
```

フロントページの続き

(72)発明者　柏木　有吾
東京都小平市上水本町５丁目20番１号　株
式会社日立製作所半導体事業部内